



# **ESCUELA POLITECNICA NACIONAL**



## **FACULTAD DE INGENIERÍA DE SISTEMAS**

**Fundamentos de Bases de Datos –  
GR1SW**

**GRUPO: No. 2**

**Manual de la Base de Datos del Proyecto del  
Segundo Bimestre**

**Alumnos: Ayala Bryan, Chugá Juan**

**FECHA DE ENTREGA: 25-01-2026**



## Manual de la Base de Datos del Proyecto de Fundamentos de Bases de Datos del Segundo Bimestre

### 1. Detalles generales de la Base de Datos

- Nombre del Sistema: “Sistema de gestión para franquicia de Barbería Profesional MEN’S”.
- Ambiente de desarrollo: el sistema fue desarrollado en JavaScript, en el IDE Visual Studio Code y con el gestor de bases de datos SQL Server.
- Autores: Bryan Ayala y Juan Chugá
- Fecha de Creación: 25/01/2026
- Nombre de la Base de Datos: BarberiaFranquiciaDB.

El enlace al repositorio donde se encuentra alojado el proyecto es el siguiente:

<https://github.com/juanchuga06/Proyecto2BFBaseDatos>

### 2. Descripción de la Base de Datos

La base de datos descrita en este manual fue creada para soportar el ingreso y salida de datos de un sistema de gestión de una franquicia de barbería profesional. Este sistema fue creado para ser usado en las diferentes sucursales de la franquicia MEN'S. Este sistema soporta tres tipos de usuarios: los franquiciados, los barberos y los clientes.

Los franquiciados pueden hacer cualquier tipo de acción dentro del sistema, es decir, pueden realizar búsquedas y reportes, agendar y revisar citas, revisar barberos y clientes, además de modificarlos y eliminarlos, añadir servicios y sedes nuevas de franquicias, etc. En cambio, los barberos solo pueden agendar citas y revisar las suyas, además de realizar búsquedas más simples. Por último, los clientes solo pueden reservar citas, y registrarse a sí mismos dentro del sistema, si no existían antes.

### 3. Arquitectura General de la Base de Datos

Se enlistan anotaciones con respecto a la implementación de la base de datos:

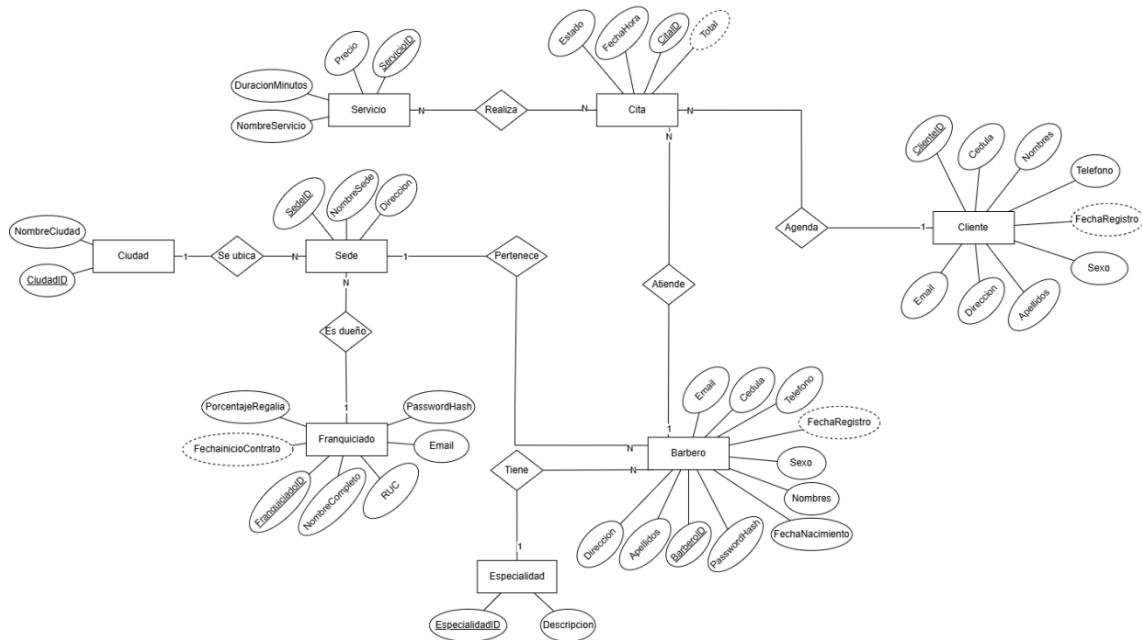
- Se usó *CamelCase* como estándar de nombres, además de que se usaron los tipos de datos estándar de SQL Server en forma de objetos TYPE.
- La base de datos fue implementada usando ‘utf-8’, de tal forma que soporte caracteres especiales del alfabeto español.
- La base de datos maneja dos tablas en forma de catálogos.
- Los dominios de la base de datos se aplicaron por medio de objetos TYPE junto a comprobaciones lógicas dentro de las definiciones de las tablas.

La base de datos se desarrolló en el motor de base de datos SQL Server 2025. Como la aplicación es WEB, se utilizó el puerto estándar 3000 para la implementación. A continuación, se exponen los modelos entidad-relación y relacional de la base de datos.

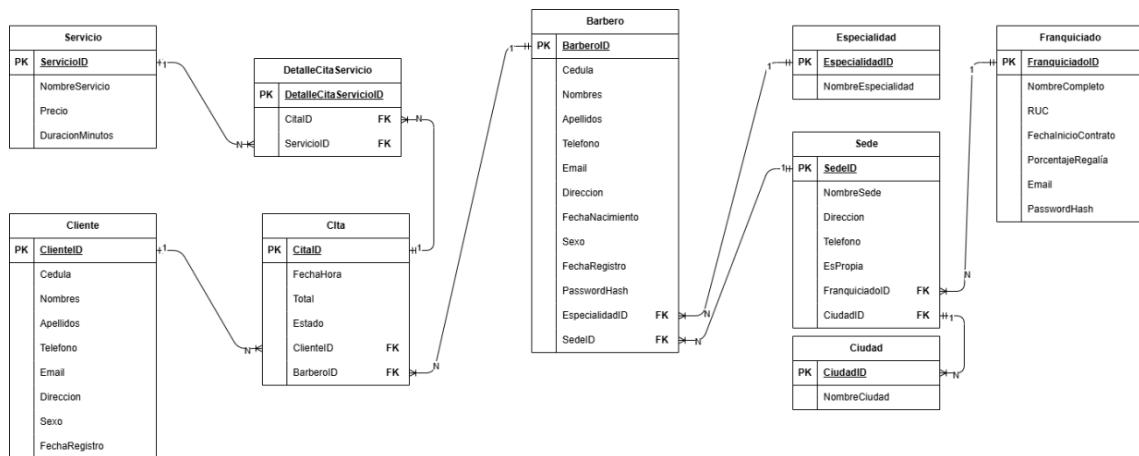


ESCUELA POLITÉCNICA NACIONAL  
FACULTAD DE INGENIERÍA DE SISTEMAS  
INGENIERÍA DE SOFTWARE

**Modelo Entidad-Relación de la Base de Datos**



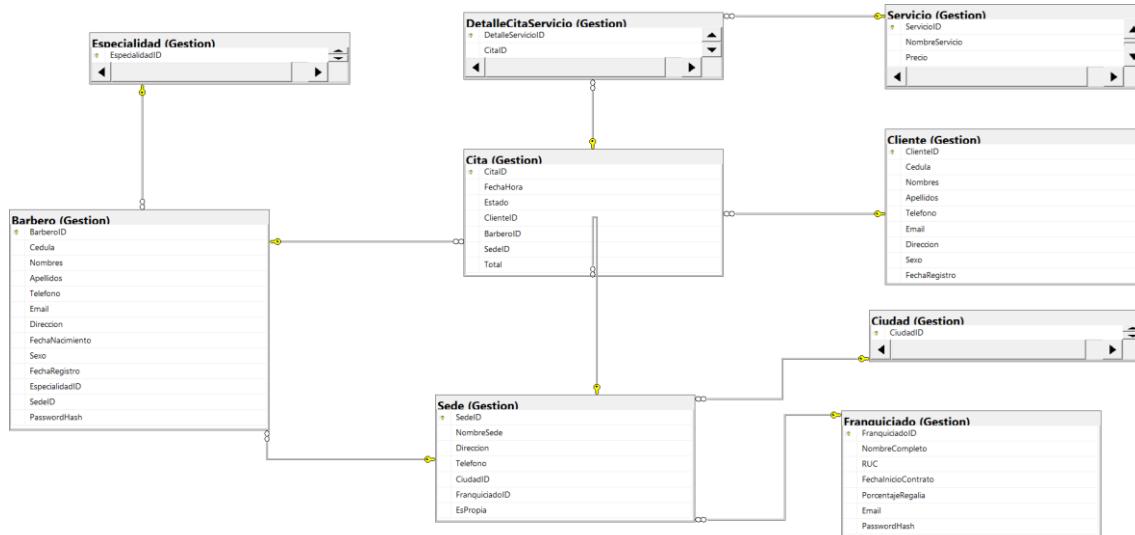
**Modelo Relacional de la Base de Datos**





ESCUELA POLITÉCNICA NACIONAL  
FACULTAD DE INGENIERÍA DE SISTEMAS  
INGENIERÍA DE SOFTWARE

### Modelo Físico de la base de Datos



### Tablas implementadas:

- Franquiciado:** representa a los socios comerciales o dueños de las franquicias, almacenando su información legal, contractual y credenciales de acceso para la gestión de sus negocios.
- Ciudad:** contiene el catálogo geográfico de las ciudades donde la marca tiene presencia operativa, sirviendo para ubicar las sedes.
- Sede:** representa el local físico o establecimiento comercial donde se prestan los servicios, vinculando su ubicación y al franquiciado responsable de su administración.
- Especialidad:** cataloga las distintas áreas de competencia técnica (como corte clásico, colorimetría o estilo urbano) para clasificar el perfil profesional de los barberos.
- Barbero:** representa al personal profesional encargado de ejecutar los servicios, estando asignado a una sede específica y poseyendo una especialidad técnica definida.
- Cliente:** almacena la información personal, de contacto y demográfica de los usuarios finales que solicitan y reciben los servicios de la barbería.
- Servicio:** define el portafolio de productos intangibles ofrecidos por la franquicia (como cortes, afeitados o tratamientos), estableciendo su precio oficial y duración estimada.
- Cita:** representa la transacción central del sistema que agenda la interacción entre un cliente y un barbero en una fecha, hora y sede determinadas, controlando además el estado de la atención (pendiente, atendida, cancelada).
- DetalleCitaServicio:** actúa como una tabla de desglose que permite vincular múltiples servicios a una sola cita, detallando exactamente qué procedimientos se realizaron en una visita específica.

### 4. Esquema

El esquema que se usa en la base de datos tiene por nombre Gestion, por tanto, el resto de los objetos que se describen en este manual están incluidos en el esquema Gestion.



## 5. Descripción de los Tipos o Dominios

En SQL Server, no existen los dominios de forma explícita, en cambio, los dominios se traducen a tipos definidos por el usuario y cláusulas CHECK o limitantes al momento de definir los atributos en las tablas. A continuación, se definen los tipos implementados en la base de datos:

	Esquema	Nombre_Tipo	Basado_En	Longitud_Max_Byes	Precision	Escala	Permite_Nulos
1	Gestion	D_Booleano	bit	1	1	0	0
2	Gestion	D_Cantidad	int	4	10	0	0
3	Gestion	D_Dinero	decimal	9	10	2	0
4	Gestion	D_Direccion	nvarchar	300	0	0	1
5	Gestion	D_Email	nvarchar	200	0	0	1
6	Gestion	D_Estado	nvarchar	40	0	0	0
7	Gestion	D_Fecha	date	3	10	0	1
8	Gestion	D_FechaHora	datetime	8	23	3	0
9	Gestion	D_ID	int	4	10	0	0
10	Gestion	D_Identificacion	nvarchar	26	0	0	0
11	Gestion	D_Nombre	nvarchar	200	0	0	0
12	Gestion	D_Porcentaje	decimal	5	5	2	0
13	Gestion	D_Sexo	char	1	0	0	1
14	Gestion	D_Teléfono	nvarchar	20	0	0	1
15	Gestion	D_TextoCorto	nvarchar	100	0	0	0

- **Gestion.D\_ID:** define un número entero no nulo, utilizado como estándar para todos los identificadores únicos (Claves Primarias) y referencias (Claves Foráneas) en el sistema.
- **Gestion.D\_Identificacion:** establece una cadena de caracteres de longitud 13, diseñada para almacenar tanto cédulas de identidad (10 dígitos) como RUCs (13 dígitos) bajo un mismo formato.
- **Gestion.D\_Nombre:** representa una cadena de texto de hasta 100 caracteres, destinada a almacenar nombres completos de personas, nombres de sedes o razones sociales extensas.
- **Gestion.D\_TextoCorto:** define un texto breve de hasta 50 caracteres, ideal para descriptores sencillos como nombres de ciudades o títulos de especialidades.
- **Gestion.D\_Direccion:** establece un campo de texto de hasta 150 caracteres que permite valores nulos, utilizado para registrar la ubicación física detallada de sedes, franquiciados o clientes.
- **Gestion.D\_Email:** representa una cadena de texto para direcciones de correo electrónico, permitiendo valores nulos en caso de que el dato no esté disponible.
- **Gestion.D\_Teléfono:** define una cadena de 10 caracteres para almacenar números de teléfono convencional o celular, estandarizando la longitud de contacto.
- **Gestion.D\_Estado:** establece un texto de hasta 20 caracteres para controlar el ciclo de vida de los procesos, limitando los valores a estados específicos como 'Pendiente', 'Atendida' o 'Cancelada'.
- **Gestion.D\_Dinero:** define un valor numérico decimal con una precisión de dos dígitos (centavos), utilizado para garantizar la exactitud en precios, totales y transacciones monetarias.
- **Gestion.D\_Cantidad:** representa un número entero, utilizado para cuantificar unidades indivisibles o medir duraciones en minutos enteros.
- **Gestion.D\_Porcentaje:** establece un valor decimal diseñado para almacenar tasas o comisiones (como las regalías), permitiendo cálculos fraccionarios precisos.



- **Gestion.D\_Fecha:** define un tipo de dato temporal que almacena únicamente el día, mes y año, utilizado para fechas de nacimiento o inicio de contratos.
- **Gestion.D\_FechaHora:** representa un registro temporal completo que incluye fecha y hora exacta, fundamental para el agendamiento preciso de las citas.
- **Gestion.D\_Sexo:** establece un campo de un solo carácter (CHAR 1) para clasificar el género biológico mediante códigos estandarizados ('M' o 'F').
- **Gestion.D\_Booleano:** define un valor binario (BIT) que actúa como un interruptor lógico (Sí/No), utilizado por ejemplo para indicar si una sede es propia o franquiciada.

## 6. Descripción de las Tablas

### 6.1 Tabla Franquiciado

Almacena la información legal y credenciales de acceso de los socios comerciales que adquieren los derechos de uso de la marca.

	Name	Owner	Type	Created_datetime							
1	Franquiciado	dbo	user table	2026-01-24 18:46:47.890							
	Column_name	Type	Computed	Length	Prec	Scale	Nullable	TrimTrailingBlanks	FixedLenNullInSource	Collation	
1	FranquiciadoID	D_ID	no	4	10	0	no	(n/a)	(n/a)	NULL	
2	NombreCompleto	D_Nombre	no	200			no	(n/a)	(n/a)	Modern_Spanish_CI_AS	
3	RUC	D_Identificacion	no	26			no	(n/a)	(n/a)	Modern_Spanish_CI_AS	
4	FechaInicioContrato	D_Fecha	no	3	10	0	no	(n/a)	(n/a)	NULL	
5	PorcentajeRegalia	D_Porcentaje	no	5	5	2	no	(n/a)	(n/a)	NULL	
6	Email	nvarchar	no	200			yes	(n/a)	(n/a)	Modern_Spanish_CI_AS	
7	PasswordHash	nvarchar	no	510			yes	(n/a)	(n/a)	Modern_Spanish_CI_AS	

#### Campos:

- FranquiciadoID — Gestion.D\_ID, autoincremental, no nulo. PK. Identificador único del franquiciado.
- NombreCompleto — Gestion.D\_Nombre, no nulo. Nombre completo o razón social del franquiciado.
- RUC — Gestion.D\_Identificacion, único, permite nulos según definición del tipo (aunque lógicamente debería ser llenado). Registro Único de Contribuyentes.
- FechaInicioContrato — Gestion.D\_Fecha, no nulo, valor por defecto fecha actual. Fecha de inicio de la relación comercial.
- PorcentajeRegalia — Gestion.D\_Porcentaje, valor por defecto 5.00. Porcentaje acordado de regalías sobre ventas.
- Email — NVARCHAR(100), permite nulos. Correo electrónico utilizado para autenticación.
- PasswordHash — NVARCHAR(255), permite nulos. Hash de la contraseña para el acceso al sistema.

#### Claves y restricciones:

- PK: FranquiciadoID.
- UNIQUE: RUC.
- Defaults: FechaInicioContrato (GETDATE()), PorcentajeRegalia (5.00).



**ESCUELA POLITÉCNICA NACIONAL**  
**FACULTAD DE INGENIERÍA DE SISTEMAS**  
**INGENIERÍA DE SOFTWARE**

## 6.2 Tabla Ciudad

Catálogo geográfico simple que lista las ciudades donde la franquicia tiene operaciones o sedes activas.

	Name	Owner	Type	Created_datetime							
1	Ciudad	dbo	user table	2026-01-24 18:46:47.897							
	Column_name	Type	Computed	Length	Prec	Scale	Nullable	TrimTrailingBlanks	FixedLenNullInSource	Collation	
1	CiudadID	D_ID	no	4	10	0	no	(n/a)	(n/a)	NULL	
2	NombreCiudad	D_TextoCorto	no	100			no	(n/a)	(n/a)	Modern_Spanish_CI_AS	

### Campos:

- CiudadID — Gestión.D\_ID, autoincremental, no nulo. PK. Identificador único de la ciudad.
- NombreCiudad — Gestión.D\_TextoCorto, permite nulos según tipo. Nombre de la ciudad.

### Claves y restricciones:

- PK: CiudadID.

## 6.3 Tabla Sede

Registra los establecimientos físicos (locales) donde se prestan los servicios, vinculándolos a una ubicación y a un responsable.

	Name	Owner	Type	Created_datetime							
1	Sede	dbo	user table	2026-01-24 18:46:47.910							
	Column_name	Type	Computed	Length	Prec	Scale	Nullable	TrimTrailingBlanks	FixedLenNullInSource	Collation	
1	SedeID	D_ID	no	4	10	0	no	(n/a)	(n/a)	NULL	
2	NombreSede	D_Nombre	no	200			no	(n/a)	(n/a)	Modern_Spanish_CI_AS	
3	Direccion	D_Direccion	no	300			yes	(n/a)	(n/a)	Modern_Spanish_CI_AS	
4	Telefono	D_Telefono	no	20			yes	(n/a)	(n/a)	Modern_Spanish_CI_AS	
5	CiudadID	D_ID	no	4	10	0	no	(n/a)	(n/a)	NULL	
6	FranquiciadoID	D_ID	no	4	10	0	no	(n/a)	(n/a)	NULL	
7	EsPropia	D_Booleano	no	1			no	(n/a)	(n/a)	NULL	

### Campos:

- SedeID — Gestión.D\_ID, autoincremental, no nulo. PK. Identificador único de la sede.
- NombreSede — Gestión.D\_Nombre, permite nulos. Nombre comercial del local.
- Direccion — Gestión.D\_Direccion, permite nulos. Ubicación física detallada.
- Telefono — Gestión.D\_Telefono, permite nulos. Número de contacto de la sede.
- CiudadID — Gestión.D\_ID, permite nulos. FK. Referencia a la ciudad de ubicación.
- FranquiciadoID — Gestión.D\_ID, permite nulos. FK. Referencia al dueño de la franquicia.
- EsPropia — Gestión.D\_Booleano, valor por defecto 0. Indica si la sede es corporativa (1) o franquiciada (0).

### Claves y restricciones:

- PK: SedeID.
- FK: CiudadID (ref. Ciudad), FranquiciadoID (ref. Franquiciado).
- Checks: Validación de formato telefónico (inicia con 0 seguido de 9 dígitos).
- Defaults: EsPropia (0).



**ESCUELA POLITÉCNICA NACIONAL**  
**FACULTAD DE INGENIERÍA DE SISTEMAS**  
**INGENIERÍA DE SOFTWARE**

#### 6.4 Tabla Especialidad

Catálogo que define las distintas competencias técnicas o categorías profesionales de los barberos.

	Name	Owner	Type	Created_datetime					
1	Especialidad	dbo	user table	2026-01-24 18:46:47.917					
Column_name	Type	Computed	Length	Prec	Scale	Nullable	TrimTrailingBlanks	FixedLenNullInSource	Collation
1	EspecialidadID	D_ID	no	4	10	0	no	(n/a)	NULL
2	NombreEspecialidad	D_TextoCorto	no	100		no	(n/a)	(n/a)	Modern_Spanish_CI_AS

#### Campos:

- EspecialidadID — Gestión.D\_ID, autoincremental, no nulo. PK. Identificador único de la especialidad.
- NombreEspecialidad — Gestión.D\_TextoCorto, permite nulos. Descripción de la especialidad técnica.

#### Claves y restricciones:

- PK: EspecialidadID.

#### 6.5 Tabla Barbero

Contiene la información personal, profesional y de acceso del personal encargado de realizar los servicios.

	Name	Owner	Type	Created_datetime					
1	Barbero	dbo	user table	2026-01-24 18:46:47.937					
Column_name	Type	Computed	Length	Prec	Scale	Nullable	TrimTrailingBlanks	FixedLenNullInSource	Collation
1	Barberoid	D_ID	no	4	10	0	no	(n/a)	NULL
2	Cedula	D_Identificacion	no	26		no	(n/a)	(n/a)	Modern_Spanish_CI_AS
3	Nombres	D_Nombre	no	200		no	(n/a)	(n/a)	Modern_Spanish_CI_AS
4	Apellidos	D_Nombre	no	200		no	(n/a)	(n/a)	Modern_Spanish_CI_AS
5	Telefono	D_Telefono	no	20		yes	(n/a)	(n/a)	Modern_Spanish_CI_AS
6	Email	D_Email	no	200		yes	(n/a)	(n/a)	Modern_Spanish_CI_AS
7	Direccion	D_Direccion	no	300		yes	(n/a)	(n/a)	Modern_Spanish_CI_AS
8	FechaNacimiento	D_Fecha	no	3	10	0	yes	(n/a)	NULL
9	Sexo	D_Sexo	no	1		yes	no	yes	Modern_Spanish_CI_AS
10	FechaRegistro	datetime	no	8		yes	(n/a)	(n/a)	NULL
11	EspecialidadID	D_ID	no	4	10	0	no	(n/a)	NULL
12	SedeID	D_ID	no	4	10	0	no	(n/a)	NULL
13	PasswordHash	nvarchar	no	510		yes	(n/a)	(n/a)	Modern_Spanish_CI_AS

#### Campos:

- Barberoid — Gestión.D\_ID, autoincremental, no nulo. PK. Identificador único del barbero.
- Cedula — Gestión.D\_Identificacion, único, permite nulos. Documento de identidad personal.
- Nombres — Gestión.D\_Nombre, permite nulos. Nombres del barbero.
- Apellidos — Gestión.D\_Nombre, permite nulos. Apellidos del barbero.
- Telefono — Gestión.D\_Telefono, permite nulos. Número de contacto personal.
- Email — Gestión.D\_Email, permite nulos. Correo electrónico profesional.
- Direccion — Gestión.D\_Direccion, permite nulos. Domicilio del barbero.
- FechaNacimiento — Gestión.D\_Fecha, permite nulos. Fecha de nacimiento para control de edad.
- Sexo — Gestión.D\_Sexo, permite nulos. Género del barbero.
- FechaRegistro — DATETIME, valor por defecto fecha actual. Fecha de ingreso al sistema.
- EspecialidadID — Gestión.D\_ID, permite nulos. FK. Especialidad principal del barbero.
- SedeID — Gestión.D\_ID, permite nulos. FK. Sede donde labora actualmente.



**ESCUELA POLITÉCNICA NACIONAL**  
**FACULTAD DE INGENIERÍA DE SISTEMAS**  
**INGENIERÍA DE SOFTWARE**

- PasswordHash — NVARCHAR(255), permite nulos. Hash de contraseña para acceso.

**Claves y restricciones:**

- PK: BarberoID.
- FK: EspecialidadID (ref. Especialidad), SedeID (ref. Sede).
- UNIQUE: Cedula.
- Checks: Cédula (10 dígitos), Teléfono (formato local), Email (formato básico con @ y .), Sexo ('M' o 'F').
- Defaults: FechaRegistro (GETDATE()).

### 6.6 Tabla Cliente

Almacena los datos de los usuarios finales que reciben los servicios de la barbería.

1	Name	Owner	Type	Created_datetime						
	Cliente	dbo	user table	2026-01-24 18:46:47.960						
	Column_name	Type	Computed	Length	Prec	Scale	Nullable	TrimTrailingBlanks	FixedLenNullInSource	Collation
1	ClienteID	D_ID	no	4	10	0	no	(n/a)	(n/a)	NULL
2	Cedula	D_Identificacion	no	26			no	(n/a)	(n/a)	Modern_Spanish_CI_AS
3	Nombres	D_Nombre	no	200			no	(n/a)	(n/a)	Modern_Spanish_CI_AS
4	Apellidos	D_Nombre	no	200			no	(n/a)	(n/a)	Modern_Spanish_CI_AS
5	Telefono	D_Telefono	no	20			yes	(n/a)	(n/a)	Modern_Spanish_CI_AS
6	Email	D_Email	no	200			yes	(n/a)	(n/a)	Modern_Spanish_CI_AS
7	Direccion	D_Direccion	no	300			yes	(n/a)	(n/a)	Modern_Spanish_CI_AS
8	Sexo	D_Sexo	no	1			yes	no	yes	Modern_Spanish_CI_AS

**Campos:**

- ClienteID — Gestion.D\_ID, autoincremental, no nulo. PK. Identificador único del cliente.
- Cedula — Gestion.D\_Identificacion, único, permite nulos. Documento de identidad.
- Nombres — Gestion.D\_Nombre, permite nulos. Nombres del cliente.
- Apellidos — Gestion.D\_Nombre, permite nulos. Apellidos del cliente.
- Telefono — Gestion.D\_Telefono, permite nulos. Teléfono de contacto.
- Email — Gestion.D\_Email, permite nulos. Correo electrónico para notificaciones.
- Direccion — Gestion.D\_Direccion, permite nulos. Domicilio del cliente.
- Sexo — Gestion.D\_Sexo, permite nulos. Género del cliente.
- FechaRegistro — DATETIME, valor por defecto fecha actual. Fecha de creación del perfil.

**Claves y restricciones:**

- PK: ClienteID.
- UNIQUE: Cedula.
- Checks: Cédula (10 dígitos), Teléfono (formato local), Email (formato básico), Sexo ('M' o 'F').

### 6.7 Tabla Servicio

Define el catálogo de servicios ofrecidos, incluyendo sus precios base y tiempos estimados de ejecución.

1	Name	Owner	Type	Created_datetime						
	Servicio	dbo	user table	2026-01-24 18:46:47.967						
	Column_name	Type	Computed	Length	Prec	Scale	Nullable	TrimTrailingBlanks	FixedLenNullInSource	Collation
1	ServicioID	D_ID	no	4	10	0	no	(n/a)	(n/a)	NULL
2	NombreServicio	D_Nombre	no	200			no	(n/a)	(n/a)	Modern_Spanish_CI_AS
3	Precio	D_Dinero	no	9	10	2	no	(n/a)	(n/a)	NULL
4	DuracionMinutos	D_Cantidad	no	4	10	0	no	(n/a)	(n/a)	NULL



**ESCUELA POLITÉCNICA NACIONAL**  
**FACULTAD DE INGENIERÍA DE SISTEMAS**  
**INGENIERÍA DE SOFTWARE**

**Campos:**

- ServicioID — Gestión.D\_ID, autoincremental, no nulo. PK. Identificador único del servicio.
- NombreServicio — Gestión.D\_Nombre, permite nulos. Nombre comercial del servicio.
- Precio — Gestión.D\_Dinero, no nulo. Costo unitario del servicio.
- DuracionMinutos — Gestión.D\_Cantidad, no nulo. Tiempo estimado en minutos.

**Claves y restricciones:**

- PK: ServicioID.

**6.8 Tabla Cita**

Registra la transacción principal de agendamiento, vinculando cliente, barbero, sede y tiempo.

	Name	Owner	Type	Created_datetime							
1	Cita	dbo	user table	2026-01-24 18:46:47.970							
	Column_name	Type	Computed	Length	Prec	Scale	Nullable	TrimTrailingBlanks	FixedLenNullInSource	Collation	
1	CitaID	D_ID	no	4	10	0	no	(n/a)	(n/a)	NULL	
2	FechaHora	D_FechaHora	no	8			no	(n/a)	(n/a)	NULL	
3	Estado	D_Estado	no	40			no	(n/a)	(n/a)	Modern_Spanish_CI_AS	
4	ClienteID	D_ID	no	4	10	0	no	(n/a)	(n/a)	NULL	
5	BarberoID	D_ID	no	4	10	0	no	(n/a)	(n/a)	NULL	
6	SedeID	D_ID	no	4	10	0	no	(n/a)	(n/a)	NULL	
7	Total	D_Dinero	no	9	10	2	no	(n/a)	(n/a)	NULL	

**Campos:**

- CitaID — Gestión.D\_ID, autoincremental, no nulo. PK. Identificador único de la cita.
- FechaHora — Gestión.D\_FechaHora, no nulo. Momento exacto de la cita.
- Estado — Gestión.D\_Estado, valor por defecto 'Pendiente'. Estado actual del proceso.
- ClienteID — Gestión.D\_ID, permite nulos. FK. Cliente que reserva.
- BarberoID — Gestión.D\_ID, permite nulos. FK. Barbero asignado.
- SedeID — Gestión.D\_ID, permite nulos. FK. Lugar de la cita.
- Total — Gestión.D\_Dinero, valor por defecto 0.00. Monto total a pagar calculado.

**Claves y restricciones:**

- PK: CitaID.
- FK: ClienteID (ref. Cliente), BarberoID (ref. Barbero), SedeID (ref. Sede).
- Checks: Estado restringido a ('Pendiente', 'Atendida', 'Cancelada', 'No Asistida').
- Defaults: Estado ('Pendiente'), Total (0.00).

**6.9 Tabla DetalleCitaServicio**

Tabla intermedia que permite asignar uno o varios servicios específicos a una sola cita.

	Name	Owner	Type	Created_datetime							
1	DetalleCitaServicio	dbo	user table	2026-01-24 18:46:47.980							
	Column_name	Type	Computed	Length	Prec	Scale	Nullable	TrimTrailingBlanks	FixedLenNullInSource	Collation	
1	DetalleServicioID	D_ID	no	4	10	0	no	(n/a)	(n/a)	NULL	
2	CitaID	D_ID	no	4	10	0	no	(n/a)	(n/a)	NULL	
3	ServicioID	D_ID	no	4	10	0	no	(n/a)	(n/a)	NULL	

**Campos:**

- DetalleServicioID — Gestión.D\_ID, autoincremental, no nulo. PK. Identificador único del detalle.
- CitaID — Gestión.D\_ID, permite nulos. FK. Cita a la que pertenece el detalle.
- ServicioID — Gestión.D\_ID, permite nulos. FK. Servicio específico contratado.



**ESCUELA POLITÉCNICA NACIONAL**  
**FACULTAD DE INGENIERÍA DE SISTEMAS**  
**INGENIERÍA DE SOFTWARE**

**Claves y restricciones:**

- PK: DetalleServicioID.
- FK: CitaID (ref. Cita), ServicioID (ref. Servicio).

## 7. Descripción de las Relaciones entre Tablas

Se explican brevemente las relaciones que unen las tablas ya descritas del modelo:

	Nombre_Constraint_FK	Tabla_Hija (Tiene la FK)	Columna_Hija	--> SE RELACIONA CON -->	Tabla_Padre (Origen)	Columna_Padre
1	FK_Barbero_Especia_5EBF139D	Gestion.Barbero	EspecialidadID	REFERENCE	Gestion.Especialidad	EspecialidadID
2	FK_Barbero_SedeID_5FB337D6	Gestion.Barbero	SedeID	REFERENCE	Gestion.Sede	SedeID
3	FK_Cita_ClientelID_6EF57B66	Gestion.Cita	ClientelID	REFERENCE	Gestion.Cliente	ClientelID
4	FK_Cita_BarberolID_6FE99F9F	Gestion.Cita	BarberolID	REFERENCE	Gestion.Barbero	BarberolID
5	FK_Cita_SedeID_70DDC3D8	Gestion.Cita	SedeID	REFERENCE	Gestion.Sede	SedeID
6	FK_DetalleCi_Cital_73BA3083	Gestion.DetalleCitaServicio	CitalID	REFERENCE	Gestion.Cita	CitalID
7	FK_DetalleCi_Servi_74AE54BC	Gestion.DetalleCitaServicio	ServicioID	REFERENCE	Gestion.Servicio	ServicioID
8	FK_Sede_CiudadID_534D60F1	Gestion.Sede	CiudadID	REFERENCE	Gestion.Ciudad	CiudadID
9	FK_Sede_Franquicia_5441852A	Gestion.Sede	FranquiciadoID	REFERENCE	Gestion.Franquiciado	FranquiciadoID

- **FRANQUICIADO y SEDE:** relación uno a muchos; SEDE.FranquiciadoID referencia al franquiciado. Un franquiciado puede ser dueño de y administrar múltiples sedes operativas.
- **CIUDAD y SEDE:** relación uno a muchos; SEDE.CiudadID referencia a la ciudad. Una ciudad geográfica puede contener varias sedes de la franquicia distribuidas en su territorio.
- **SEDE y BARBERO:** relación uno a muchos; BARBERO.SedeID referencia a la sede. Una sede física emplea a múltiples barberos para la atención al público.
- **ESPECIALIDAD y BARBERO:** relación uno a muchos; BARBERO.EspecialidadID referencia a la especialidad. Una misma especialidad técnica (como "Corte Infantil") puede ser poseída por varios barberos distintos.
- **CLIENTE y CITA:** relación uno a muchos; CITA.ClienteID referencia al cliente. Un cliente registrado puede agendar múltiples citas a lo largo de su historial.
- **BARBERO y CITA:** relación uno a muchos; CITA.BarberolID referencia al barbero. Un barbero puede atender muchas citas distintas dentro de su agenda laboral.
- **SEDE y CITA:** relación uno a muchos; CITA.SedeID referencia a la sede. Cada cita agendada se lleva a cabo obligatoriamente en una única sede física específica.
- **CITA y DETALLECITA y SERVICIO:** relación muchos a muchos entre citas y servicios, modelada por la tabla intermedia DETALLECTASERVICIO. Una cita puede incluir varios servicios simultáneos (ej. corte y barba), y un tipo de servicio puede estar presente en múltiples citas diferentes. Es importante notar que la tabla intermedia utiliza su propia clave primaria (DetalleServicioID) en lugar de una clave compuesta para gestionar la unicidad de cada registro.



## 8. Funciones de la base de datos

Se explican brevemente las funciones que modifican las tablas ya descritas del modelo:

	Nombre_Funcion	Tipo	Definicion_SQL
1	fn_EsBarberoDisponible	SQL_SCALAR_FUNCTION	-- =====...
2	fn_BarberoPerteneceASede	SQL_SCALAR_FUNCTION	CREATE FUNCTION Gestion.fn_Barbero...
3	fn_CalcularTotalConIVA	SQL_SCALAR_FUNCTION	-- IMPORTANTE: Cálculo con JOIN (Lógic...
4	fn_CalcularEdad	SQL_SCALAR_FUNCTION	CREATE FUNCTION Gestion.fn_Calcular...

- **Gestion.fn\_EsBarberoDisponible:** Función lógica diseñada para evitar conflictos de agenda. Recibe como parámetros un identificador de barbero y una fecha/hora específica; verifica en la tabla Cita si existe algún registro activo (no cancelado) para ese barbero en ese mismo momento. Retorna un valor booleano (1 o 0) indicando si el profesional está libre para ser reservado.

```
CREATE OR ALTER FUNCTION Gestion.fn_EsBarberoDisponible(@BarberoID INT, @FechaHora DATETIME)
RETURNS BIT AS
BEGIN
    DECLARE @EstaDisponible BIT = 1;
    IF EXISTS (SELECT 1 FROM Gestion.Cita WHERE BarberoID = @BarberoID AND FechaHora = @FechaHora AND Estado <> 'Cancelada')
    | SET @EstaDisponible = 0;
    RETURN @EstaDisponible;
END;
GO
```

- **Gestion.fn\_BarberoPerteneceASede:** Función de validación operativa que asegura la coherencia geográfica de las citas. Recibe el ID de un barbero y el ID de una sede, y comprueba en la tabla Barbero si existe esa relación laboral específica. Retorna un valor booleano confirmando si el barbero efectivamente trabaja en el lugar donde se intenta agendar la cita.

```
CREATE OR ALTER FUNCTION Gestion.fn_BarberoPerteneceASede(@BarberoID INT, @SedeID INT)
RETURNS BIT AS
BEGIN
    DECLARE @Pertenece BIT = 0;
    IF EXISTS (SELECT 1 FROM Gestion.Barbero WHERE BarberoID = @BarberoID AND SedeID = @SedeID)
    | SET @Pertenece = 1;
    RETURN @Pertenece;
END;
GO
```

- **Gestion.fn\_CalcularTotalConIVA:** Función financiera encargada de automatizar la facturación. Toma el ID de una cita, suma los precios individuales de todos los servicios asociados encontrados en la tabla DetalleCitaServicio (Subtotal) y aplica automáticamente un factor de 1.15 para incluir el 15% de IVA. Retorna el monto decimal final a pagar por el cliente.

```
CREATE OR ALTER FUNCTION Gestion.fn_CalcularTotalConIVA(@CitaID INT)
RETURNS DECIMAL(10,2) AS
BEGIN
    DECLARE @Subtotal DECIMAL(10,2);
    SELECT @Subtotal = ISNULL(SUM(S.Precio), 0)
    FROM Gestion.DetalleCitaServicio D
    INNER JOIN Gestion.Servicio S ON D.ServicioID = S.ServicioID
    WHERE D.CitaID = @CitaID;
    RETURN @Subtotal * 1.15;
END;
GO
```



- **Gestion.fn\_CalcularEdad:** Función utilitaria de cálculo demográfico. Recibe una fecha de nacimiento y calcula la edad exacta en años comparándola con la fecha actual del sistema (GETDATE), ajustando el cálculo si el cumpleaños aún no ha ocurrido en el año corriente. Es utilizada principalmente para validar restricciones legales, como asegurar que los barberos sean mayores de edad.

```
CREATE OR ALTER FUNCTION Gestion.fn_CalcularEdad(@FechaNacimiento DATE)
RETURNS INT AS
BEGIN
    DECLARE @Edad INT;
    SET @Edad = DATEDIFF(YEAR, @FechaNacimiento, GETDATE()) -
                CASE WHEN (MONTH(@FechaNacimiento) > MONTH(GETDATE())) OR
                      (MONTH(@FechaNacimiento) = MONTH(GETDATE()) AND DAY(@FechaNacimiento) > DAY(GETDATE()))
                THEN 1 ELSE 0 END;
    RETURN @Edad;
END;
GO
```

## 9. Triggers de la base de datos

Se explican brevemente los disparadores o triggers que ejecutan las funciones de la sección anterior:

	Tabla_Padre	Nombre_Trigger	Definicion_SQL
1	Cita	trg_ValidarDisponibilidadCita	-- =====...
2	Cita	trg_ValidarSedeBarbero	CREATE TRIGGER Gestion.trg_ValidarSedeBarbero ON ...
3	DetalleCitaServicio	trg_MantenimientoTotalCita	CREATE TRIGGER Gestion.trg_MantenimientoTotalCita O...
4	Barbero	trg_ValidarEdadBarbero	CREATE TRIGGER Gestion.trg_ValidarEdadBarbero ON ...

- **Gestion.trg\_ValidarDisponibilidadCita:** Disparador de control de agenda que se activa automáticamente tras la inserción o actualización de registros en la tabla Cita. Su función es garantizar la exclusividad del tiempo del personal, bloqueando y revirtiendo la transacción (ROLLBACK) si detecta que el barbero seleccionado ya posee otra cita activa (no cancelada) exactamente en el mismo horario.

```
CREATE OR ALTER TRIGGER Gestion.trg_ValidarDisponibilidadCita ON Gestion.Cita AFTER INSERT, UPDATE AS
BEGIN
    SET NOCOUNT ON;
    IF EXISTS (SELECT 1 FROM inserted i WHERE (SELECT COUNT(*) FROM Gestion.Cita WHERE BarberoID = i.BarberoID AND FechaHora = i.FechaHora AND Estado <> 'Cancelada') > 1)
    BEGIN
        RAISERROR ('El barbero ya tiene una cita en ese horario.', 16, 1);
        ROLLBACK TRANSACTION;
    END
END;
GO
```

- **Gestion.trg\_ValidarSedeBarbero:** Disparador de coherencia operativa diseñado para validar la lógica geográfica al agendar. Verifica que el barbero asignado a una cita pertenezca efectivamente a la sede donde se está programando el servicio; si el sistema detecta que el barbero no trabaja en ese local, impide la creación de la cita para evitar errores de logística.

```
CREATE OR ALTER TRIGGER Gestion.trg_ValidarSedeBarbero ON Gestion.Cita AFTER INSERT, UPDATE AS
BEGIN
    SET NOCOUNT ON;
    IF EXISTS (SELECT 1 FROM inserted i WHERE NOT EXISTS (SELECT 1 FROM Gestion.Barbero WHERE BarberoID = i.BarberoID AND SedeID = i.SedeID))
    BEGIN
        RAISERROR ('El barbero no trabaja en la sede de la cita.', 16, 1);
        ROLLBACK TRANSACTION;
    END
END;
GO
```

- **Gestion.trg\_MantenimientoTotalCita:** Disparador de automatización financiera asociado a la tabla DetalleCitaServicio. Se ejecuta en tiempo real cada vez que se agregan,



ESCUELA POLITÉCNICA NACIONAL  
FACULTAD DE INGENIERÍA DE SISTEMAS  
INGENIERÍA DE SOFTWARE

eliminan o modifican servicios de una cita, encargándose de recalcular y actualizar automáticamente el campo Total en la tabla principal Cita (usando la función de cálculo con IVA), asegurando que el precio a cobrar siempre esté sincronizado con los servicios contratados.

```
CREATE OR ALTER TRIGGER Gestion.trg_MantenimientoTotalCita ON Gestion.DetalleCitaServicio AFTER INSERT, DELETE, UPDATE AS
BEGIN
    SET NOCOUNT ON;
    DECLARE @CitasAfectadas TABLE (CitaID INT);
    INSERT INTO @CitasAfectadas (CitaID) SELECT DISTINCT CitaID FROM inserted UNION SELECT DISTINCT CitaID FROM deleted;
    UPDATE C SET C.Total = Gestion.fn_CalcularTotalConIVA(C.CitaID) FROM Gestion.Cita C INNER JOIN @CitasAfectadas A ON C.CitaID = A.CitaID;
END;
GO
```

- **Gestion.trg\_ValidarEdadBarbero:** Disparador de cumplimiento legal que monitorea la tabla Barbero. Al intentar registrar un nuevo empleado o modificar su fecha de nacimiento, utiliza la función de cálculo de edad para verificar que la persona sea mayor de 18 años; si no cumple este requisito, el sistema rechaza la operación inmediatamente para evitar la contratación de menores.

```
CREATE OR ALTER TRIGGER Gestion.trg_ValidarEdadBarbero ON Gestion.Barbero AFTER INSERT, UPDATE AS
BEGIN
    SET NOCOUNT ON;
    IF EXISTS (SELECT 1 FROM inserted WHERE Gestion.fn_CalcularEdad(FechaNacimiento) < 18)
    BEGIN
        RAISERROR ('El barbero debe ser mayor de 18 años.', 16, 1);
        ROLLBACK TRANSACTION;
    END
END;
GO
```

## 10. Procedimientos almacenados de la base de datos

Se han implementado diversos procedimientos almacenados para encapsular la lógica de manipulación de datos, garantizando seguridad, integridad y reusabilidad en el código. Estos objetos se dividen en dos categorías principales según su función en el sistema.

**10.1. Procedimientos de Mantenimiento (CRUD):** Este grupo de procedimientos gestiona las operaciones fundamentales (Crear, Leer, Actualizar, Eliminar) de las entidades maestras y catálogos del sistema. Su diseño unificado utiliza un parámetro `@Accion` para seleccionar la operación deseada.

- **Gestion.sp\_CRUD\_Barbero:** Este procedimiento es el encargado de administrar el ciclo de vida de los barberos en el sistema. Permite registrar nuevos profesionales, actualizar su información personal o eliminarlos. Su lógica incluye una validación de integridad referencial crítica: antes de procesar una eliminación (`@Accion = 'D'`), verifica si el barbero tiene citas históricas asignadas; de ser así, el sistema bloquea la operación y notifica el error para evitar inconsistencias en los reportes financieros.

```
CREATE OR ALTER PROCEDURE Gestion.sp_CRUD_Barbero @ID Gestion.D_ID = NULL, @Cedula Gestion.D_Identificacion = NULL,
@Nombres Gestion.D_Nombre = NULL, @Apellidos Gestion.D_Nombre = NULL, @Telefono Gestion.D_Telefono = NULL, @Email Gestion.D_Email = NULL,
@EspecialidadID Gestion.D_ID = NULL, @SedeID Gestion.D_ID = NULL, @FechaNacimiento DATE = NULL, @Accion CHAR(1) AS
BEGIN
    IF @Accion = 'I' INSERT INTO Gestion.Barbero (Cedula, Nombres, Apellidos, Telefono, Email, EspecialidadID, SedeID,
    FechaNacimiento) VALUES (@Cedula, @Nombres, @Apellidos, @Telefono, @Email, @EspecialidadID, @SedeID,
    @FechaNacimiento);
    ELSE IF @Accion = 'U' UPDATE Gestion.Barbero SET Telefono = @Telefono, Email = @Email, EspecialidadID =
    @EspecialidadID, SedeID = @SedeID WHERE BarberoID = @ID;
    ELSE IF @Accion = 'D' BEGIN
        IF NOT EXISTS (SELECT 1 FROM Gestion.Cita WHERE BarberoID = @ID) DELETE FROM Gestion.Barbero WHERE BarberoID =
        @ID;
        ELSE RAISERROR ('No se puede eliminar: Barbero tiene citas.', 16, 1);
    END
END;
GO
```



ESCUELA POLITÉCNICA NACIONAL  
FACULTAD DE INGENIERÍA DE SISTEMAS  
INGENIERÍA DE SOFTWARE

- **Gestion.sp\_CRUD\_Cliente:** Este procedimiento administra la base de datos de los usuarios finales del sistema. Facilita el registro de nuevos perfiles, la actualización de datos de contacto y la eliminación de registros, siempre verificando que no existan citas vinculadas al cliente para mantener la integridad de la información histórica y financiera.

```
CREATE OR ALTER PROCEDURE Gestion.sp_CRUD_Cliente @ID Gestion.D_ID = NULL, @Cedula Gestion.D_Identificacion = NULL,
@Nombres Gestion.D_Nombre = NULL, @Apellidos Gestion.D_Nombre = NULL, @Telefono Gestion.D_Telefono = NULL, @Email Gestion.
D_Email = NULL, @Direccion Gestion.D_Direccion = NULL, @Sexo Gestion.D_Sexo = NULL, @Accion CHAR(1) AS
BEGIN
    IF @Accion = 'I' INSERT INTO Gestion.Cliente (Cedula, Nombres, Apellidos, Telefono, Email, Direccion, Sexo) VALUES
        (@Cedula, @Nombres, @Apellidos, @Telefono, @Email, @Direccion, @Sexo);
    ELSE IF @Accion = 'U' UPDATE Gestion.Cliente SET Telefono = @Telefono, Email = @Email, Direccion = @Direccion WHERE
        ClienteID = @ID;
    ELSE IF @Accion = 'D' BEGIN
        IF NOT EXISTS (SELECT 1 FROM Gestion.Cita WHERE ClienteID = @ID) DELETE FROM Gestion.Cliente WHERE ClienteID =
            @ID;
        ELSE RAISERROR ('No se puede eliminar: Cliente tiene citas.', 16, 1);
    END
END;
GO
```

- **Gestion.sp\_CRUD\_Sede:** Este procedimiento es el responsable de administrar la información de los establecimientos físicos donde se prestan los servicios de barbería. Facilita la creación de nuevas sedes asociándolas correctamente a su ciudad y franquiciado correspondiente, permitiendo además actualizar datos operativos como el nombre comercial, la dirección exacta y el teléfono de contacto de cada local.

```
CREATE OR ALTER PROCEDURE Gestion.sp_CRUD_Sede @ID Gestion.D_ID = NULL, @Nombre Gestion.D_Nombre = NULL, @Direccion
Gestion.D_Direccion = NULL, @Telefono Gestion.D_Telefono = NULL, @CiudadID Gestion.D_ID = NULL, @FranquiciadoID Gestion.
D_ID = NULL, @Accion CHAR(1) AS
BEGIN
    IF @Accion = 'I' INSERT INTO Gestion.Sede (NombreSede, Direccion, Telefono, CiudadID, FranquiciadoID) VALUES
        (@Nombre, @Direccion, @Telefono, @CiudadID, @FranquiciadoID);
    ELSE IF @Accion = 'U' UPDATE Gestion.Sede SET NombreSede = @Nombre, Direccion = @Direccion, Telefono = @Telefono
        WHERE SedeID = @ID;
END;
GO
```

- **Gestion.sp\_CU\_Servicio:** Este procedimiento se encarga de gestionar el catálogo de productos intangibles de la barbería. Permite la creación de nuevos servicios definiendo su nombre comercial, precio base y duración estimada en minutos, además de facilitar la actualización de estos valores conforme cambie la oferta operativa de la franquicia.

```
CREATE OR ALTER PROCEDURE Gestion.sp_CU_Servicio @ID Gestion.D_ID = NULL, @Nombre Gestion.D_Nombre, @Precio Gestion.
D_Dinero, @Duracion Gestion.D_Cantidad, @Accion CHAR(1) AS
BEGIN
    IF @Accion = 'I' INSERT INTO Gestion.Servicio (NombreServicio, Precio, DuracionMinutos) VALUES (@Nombre, @Precio,
        @Duracion);
    ELSE IF @Accion = 'U' UPDATE Gestion.Servicio SET NombreServicio = @Nombre, Precio = @Precio, DuracionMinutos =
        @Duracion WHERE ServicioID = @ID;
END;
GO
```

- **Gestion.sp\_CU\_Franquiciado:** Administra el registro de los socios comerciales de la marca. Permite la inserción de nuevos franquiciados almacenando su nombre, RUC y porcentaje de regalía acordado, facilitando también la actualización de estos datos según los contratos vigentes.

```
CREATE OR ALTER PROCEDURE Gestion.sp_CU_Franquiciado @ID Gestion.D_ID = NULL, @Nombre Gestion.D_Nombre, @RUC Gestion.
D_Identificacion, @Regalia Gestion.D_Porcentaje, @Accion CHAR(1) AS
BEGIN
    IF @Accion = 'I' INSERT INTO Gestion.Franquiciado (NombreCompleto, RUC, PorcentajeRegalia) VALUES (@Nombre, @RUC,
        @Regalia);
    ELSE IF @Accion = 'U' UPDATE Gestion.Franquiciado SET NombreCompleto = @Nombre, PorcentajeRegalia = @Regalia WHERE
        FranquiciadoID = @ID;
END;
GO
```



ESCUELA POLITÉCNICA NACIONAL  
FACULTAD DE INGENIERÍA DE SISTEMAS  
INGENIERÍA DE SOFTWARE

- **Gestion.sp\_CU\_Ciudad:** Este procedimiento gestiona el catálogo de ubicaciones geográficas donde la marca opera. Permite crear, modificar o eliminar ciudades, incorporando una validación preventiva que impide el borrado de una ubicación si esta ya tiene sedes operativas registradas.

```
CREATE OR ALTER PROCEDURE Gestion.sp_CU_Ciudad @ID Gestion.D_ID = NULL, @Nombre Gestion.D_TextoCorto, @Acción CHAR(1) AS
BEGIN
    IF @Acción = 'I' INSERT INTO Gestion.Ciudad (NombreCiudad) VALUES (@Nombre);
    ELSE IF @Acción = 'U' UPDATE Gestion.Ciudad SET NombreCiudad = @Nombre WHERE CiudadID = @ID;
    ELSE IF @Acción = 'D' BEGIN
        IF NOT EXISTS (SELECT 1 FROM Gestion.Sede WHERE CiudadID = @ID) DELETE FROM Gestion.Ciudad WHERE CiudadID = @ID;
        ELSE RAISERROR ('No se puede eliminar: Ciudad tiene sedes.', 16, 1);
    END
END;
GO
```

- **Gestion.sp\_CU\_Especialidad:** Gestiona las diferentes áreas de competencia técnica para clasificar al personal. Permite insertar nuevas especialidades y actualizar sus nombres comerciales, bloqueando la eliminación de cualquier categoría que esté asignada actualmente a un barbero en activo.

```
CREATE OR ALTER PROCEDURE Gestion.sp_CU_Especialidad @ID Gestion.D_ID = NULL, @Nombre Gestion.D_TextoCorto, @Acción CHAR(1) AS
BEGIN
    IF @Acción = 'I' INSERT INTO Gestion.Especialidad (NombreEspecialidad) VALUES (@Nombre);
    ELSE IF @Acción = 'U' UPDATE Gestion.Especialidad SET NombreEspecialidad = @Nombre WHERE EspecialidadID = @ID;
    ELSE IF @Acción = 'D' BEGIN
        IF NOT EXISTS (SELECT 1 FROM Gestion.Barbero WHERE EspecialidadID = @ID) DELETE FROM Gestion.Especialidad WHERE EspecialidadID = @ID;
        ELSE RAISERROR ('No se puede eliminar: Especialidad tiene barberos.', 16, 1);
    END
END;
GO
```

**10.2. Procedimientos Transaccionales (Gestión de Citas):** Esta sección detalla los procedimientos que ejecutan la lógica operativa central de la barbería, permitiendo gestionar el flujo de agendamiento, la asignación de servicios y el control de estados de las citas.

- **Gestion.sp\_InsertarCita:** Este procedimiento es el encargado de iniciar el proceso de agendamiento en el sistema. Registra una nueva cita vinculando la fecha, el cliente, el barbero y la sede, asignando automáticamente el estado 'Pendiente' y devolviendo el ID de la transacción para continuar con la adición de servicios.

```
CREATE OR ALTER PROCEDURE Gestion.sp_InsertarCita @FechaHora Gestion.D_FechaHora, @ClienteID Gestion.D_ID, @BarberoID Gestion.D_ID, @SedeID Gestion.D_ID AS
BEGIN
    INSERT INTO Gestion.Cita (FechaHora, ClienteID, BarberoID, SedeID, Estado, Total) VALUES (@FechaHora, @ClienteID, @BarberoID, @SedeID, 'Pendiente', 0.00);
    SELECT SCOPE_IDENTITY() AS NuevaCitaID;
END;
GO
```

- **Gestion.sp\_ActualizarCita:** Este procedimiento permite modificar los datos de una cita ya registrada, incluyendo cambios en la fecha, hora, cliente, barbero o sede. Es una herramienta esencial para el control operativo, ya que permite actualizar el estado de la cita conforme avanza el proceso de atención en el local.

```
CREATE OR ALTER PROCEDURE Gestion.sp_ActualizarCita @CitaID Gestion.D_ID, @FechaHora Gestion.D_FechaHora, @ClienteID Gestion.D_ID, @BarberoID Gestion.D_ID, @SedeID Gestion.D_ID, @Estado Gestion.D_Estado AS
BEGIN
    UPDATE Gestion.Cita SET FechaHora = @FechaHora, ClienteID = @ClienteID, BarberoID = @BarberoID, SedeID = @SedeID, Estado = @Estado WHERE CitaID = @CitaID;
END;
GO
```



- **Gestion.sp\_InsertarDetalleCita:** Este procedimiento permite añadir servicios individuales a una cita ya existente. Es fundamental para el funcionamiento de la tabla intermedia, ya que vincula el identificador de la cita con el servicio seleccionado, permitiendo que una sola visita a la barbería incluya múltiples procedimientos de forma organizada.

```
CREATE OR ALTER PROCEDURE Gestion.sp_InsertarDetalleCita @CitaID Gestion.D_ID, @ServicioID Gestion.D_ID AS
BEGIN
    INSERT INTO Gestion.DetalleCitaServicio (CitaID, ServicioID) VALUES (@CitaID, @ServicioID);
END;
GO
```

- **Gestion.sp\_ActualizarDetalleCita:** Este procedimiento permite modificar un servicio específico que ya ha sido asignado previamente a una cita. Su función es facilitar la corrección de errores en la selección de servicios o realizar cambios de último momento solicitados por el cliente, manteniendo siempre la precisión en el desglose de la atención.

```
CREATE OR ALTER PROCEDURE Gestion.sp_ActualizarDetalleCita @DetalleServicioID Gestion.D_ID, @ServicioID Gestion.D_ID AS
BEGIN
    UPDATE Gestion.DetalleCitaServicio SET ServicioID = @ServicioID WHERE DetalleServicioID = @DetalleServicioID;
END;
GO
```

- **Gestion.sp\_CancelarCita:** Este procedimiento se encarga de cambiar el estado de una cita a 'Cancelada' utilizando su identificador único. A diferencia de una eliminación física, este método mantiene el registro en la base de datos para fines de auditoría y análisis, permitiendo rastrear las citas anuladas sin perder la integridad histórica del sistema de agendamiento.

```
CREATE OR ALTER PROCEDURE Gestion.sp_CancelarCita @CitaID Gestion.D_ID AS
BEGIN
    UPDATE Gestion.Cita SET Estado = 'Cancelada' WHERE CitaID = @CitaID;
END;
GO
```

## 11. Cursores de la base de datos

La base de datos incluye solo un cursor, que está contenido en el último procedimiento almacenado y se describe a continuación:

- **Gestion.sp\_ProcesarCitasNoAsistidas:** Procedimiento de mantenimiento automatizado que implementa el cursor cur\_CitasVencidas para la depuración de la agenda. Su función es recorrer secuencialmente (registro por registro) todas las citas que permanecen en estado 'Pendiente' pero cuya fecha programada es anterior al momento actual (citas vencidas). Para cada caso identificado, el cursor ejecuta una actualización individual cambiando el estado a 'No Asistida', permitiendo así cerrar los ciclos de atención y clasificar el ausentismo de los clientes sin intervención manual.

```
CREATE OR ALTER PROCEDURE Gestion.sp_ProcesarCitasNoAsistidas AS
BEGIN
    SET NOCOUNT ON;
    DECLARE @CitaID_Actual INT, @Contador INT = 0;
    DECLARE cur_CitasVencidas CURSOR FOR SELECT CitaID FROM Gestion.Cita WHERE Estado = 'Pendiente' AND FechaHora < GETDATE();
    OPEN cur_CitasVencidas;
    FETCH NEXT FROM cur_CitasVencidas INTO @CitaID_Actual;
    WHILE @@FETCH_STATUS = 0
    BEGIN
        UPDATE Gestion.Cita SET Estado = 'No Asistida' WHERE CitaID = @CitaID_Actual;
        SET @Contador = @Contador + 1;
        FETCH NEXT FROM cur_CitasVencidas INTO @CitaID_Actual;
    END
    CLOSE cur_CitasVencidas;
    DEALLOCATE cur_CitasVencidas;
    PRINT 'Citas marcadas como No Asistidas: ' + CAST(@Contador AS NVARCHAR(10));
END;
GO
```



## 12. Vistas de la base de datos

Se han implementado 20 vistas predefinidas que permiten extraer información estratégica de la base de datos sin necesidad de ejecutar consultas complejas. Estas vistas facilitan la generación de reportes automáticos y la toma de decisiones para los franquiciados.

### 12.1. Bloque 1: Análisis Financiero

- **Gestion.v\_ReporteRegaliasMensuales:** Esta vista permite calcular el monto exacto que cada franquiciado debe pagar a la casa matriz por concepto de regalías. Utiliza la venta bruta total de las citas atendidas y el porcentaje de regalía estipulado en el contrato de cada socio para generar el reporte de liquidación mensual.

```
CREATE OR ALTER VIEW Gestion.v_ReporteRegaliasMensuales AS
SELECT f.NombreCompleto AS "Franquiciado", COUNT(c.CitaID) AS "Citas Atendidas", SUM(c.Total) AS "Venta Bruta", f.PorcentajeRegalia AS "% Regalía", SUM(c.Total * f.PorcentajeRegalia / 100.0) AS "Total A Pagar"
FROM Gestion.Cita c INNER JOIN Gestion.Sede s ON s.SedeID = c.SedeID INNER JOIN Gestion.Franquiciado f ON f.FranquiciadoID = s.FranquiciadoID
WHERE c.Estado = 'Atendida' GROUP BY f.NombreCompleto, f.PorcentajeRegalia;
GO
```

- **Gestion.v\_IngresosPorCiudad:** Facilita la comparación del rendimiento económico entre las distintas ubicaciones geográficas. Agrupa los ingresos generados por todas las sedes dentro de una misma ciudad, permitiendo identificar qué mercados son los más rentables para la franquicia.

```
CREATE OR ALTER VIEW Gestion.v_IngresosPorCiudad AS
SELECT TOP 100 PERCENT ciu.NombreCiudad AS "Ciudad", ISNULL(SUM(c.Total), 0) AS "Ingreso Generado"
FROM Gestion.Ciudad ciu LEFT JOIN Gestion.Sede s ON s.CiudadID = ciu.CiudadID LEFT JOIN Gestion.Cita c ON c.SedeID = s.SedeID AND c.Estado = 'Atendida'
GROUP BY ciu.NombreCiudad ORDER BY "Ingreso Generado" DESC;
GO
```

- **Gestion.v\_TicketPromedioPorSede:** Calcula el gasto medio que realiza un cliente cada vez que acude a una sede específica. Es una métrica de rendimiento clave (KPI) que ayuda a evaluar la capacidad de cada local para vender servicios adicionales o de mayor valor.

```
CREATE OR ALTER VIEW Gestion.v_TicketPromedioPorSede AS
SELECT TOP 100 PERCENT s.NombreSede AS "Sede", COUNT(c.CitaID) AS "Total Citas", AVG(c.Total) AS "Ticket Promedio"
FROM Gestion.Sede s INNER JOIN Gestion.Cita c ON s.SedeID = c.SedeID WHERE c.Estado = 'Atendida'
GROUP BY s.NombreSede ORDER BY "Ticket Promedio" DESC;
GO
```

- **Gestion.v\_SedesBajoRendimiento:** Actúa como un sistema de alerta temprana al listar las sedes cuya venta total acumulada es inferior a un umbral crítico de \$500. Permite a la administración identificar locales que requieren planes de mejora operativa o de marketing.

```
CREATE OR ALTER VIEW Gestion.v_SedesBajoRendimiento AS
SELECT s.NombreSede, ISNULL(SUM(c.Total), 0) AS "Venta Total Historica"
FROM Gestion.Sede s LEFT JOIN Gestion.Cita c ON s.SedeID = c.SedeID AND c.Estado = 'Atendida'
GROUP BY s.NombreSede HAVING ISNULL(SUM(c.Total), 0) < 500;
GO
```

### 12.2. Bloque 2: Productividad Operativa

- **Gestion.v\_TopBarberosDelMes:** Esta vista genera un ranking de los tres profesionales que han logrado la mayor recaudación económica durante el mes en curso. Es una herramienta de motivación y evaluación de desempeño que filtra únicamente las citas con estado 'Atendida' para asegurar la precisión de los datos.



ESCUELA POLITÉCNICA NACIONAL  
FACULTAD DE INGENIERÍA DE SISTEMAS  
INGENIERÍA DE SOFTWARE

```
CREATE OR ALTER VIEW Gestion.v_TopBarberosDelMes AS
SELECT TOP 3 b.Nombres + ' ' + b.Apellidos AS "Barbero", SUM(c.Total) AS "Recaudado Este Mes"
FROM Gestion.Barbero b INNER JOIN Gestion.Cita c ON b.BarberoID = c.BarberoID
WHERE c.Estado = 'Atendida' AND MONTH(c.FechaHora) = MONTH(GETDATE()) AND YEAR(c.FechaHora) = YEAR(GETDATE())
GROUP BY b.Nombres, b.Apellidos ORDER BY "Recaudado Este Mes" DESC;
GO
```

- **Gestion.v\_EficienciaCitas:** Permite medir la fiabilidad de la agenda de cada barbero al mostrar el total de citas asignadas frente al número de cancelaciones. Calcula automáticamente el porcentaje de cancelación por profesional, lo que ayuda a identificar patrones que podrían afectar la rentabilidad del local.

```
CREATE OR ALTER VIEW Gestion.v_EficienciaCitas AS
SELECT b.Nombres AS "Barbero", COUNT(c.CitaID) AS "Citas Totales", SUM(CASE WHEN c.Estado = 'Cancelada' THEN 1 ELSE 0 END) AS "Canceladas",
CAST(SUM(CASE WHEN c.Estado = 'Cancelada' THEN 1 ELSE 0 END) * 100.0 / COUNT(c.CitaID) AS DECIMAL(5,2)) AS "% Cancelacion"
FROM Gestion.Barbero b INNER JOIN Gestion.Cita c ON b.BarberoID = c.BarberoID GROUP BY b.Nombres;
GO
```

- **Gestion.v\_BarberosMultifaceticos:** Identifica a los colaboradores con mayor versatilidad técnica al listar a aquellos que han ejecutado más de dos tipos de servicios distintos. Es útil para la gestión de talento, permitiendo asignar personal capacitado a sedes con demandas de servicios complejos.

```
CREATE OR ALTER VIEW Gestion.v_BarberosMultifaceticos AS
SELECT b.Nombres, COUNT(DISTINCT d.ServicioID) AS "Tipos Servicios Distintos"
FROM Gestion.Barbero b INNER JOIN Gestion.Cita c ON b.BarberoID = c.BarberoID INNER JOIN Gestion.DetalleCitaServicio d ON
c.CitaID = d.CitaID
GROUP BY b.Nombres HAVING COUNT(DISTINCT d.ServicioID) > 2;
GO
```

- **Gestion.v\_AntiguedadPersonal:** Calcula el tiempo de permanencia de cada barbero en la franquicia en días. Esta métrica es esencial para la gestión de recursos humanos y para la planificación de incentivos basados en la lealtad y experiencia del personal.

```
CREATE OR ALTER VIEW Gestion.v_AntiguedadPersonal AS
SELECT Nombres + ' ' + Apellidos AS "Personal", FechaRegistro, DATEDIFF(DAY, FechaRegistro, GETDATE()) AS "Dias Trabajando" FROM Gestion.Barbero;
GO
```

### 12.3. Bloque 3: Inteligencia de Clientes

- **Gestion.v\_ClientesVIP:** Filtra y ordena a los clientes con mayor valor de vida (LTV), identificando a aquellos cuyo gasto histórico supera los \$50. Permite a la franquicia diseñar programas de lealtad o beneficios exclusivos para sus usuarios más rentables.

```
CREATE OR ALTER VIEW Gestion.v_ClientesVIP AS
SELECT TOP 100 PERCENT cli.Nombres + ' ' + cli.Apellidos AS "Cliente VIP", SUM(c.Total) AS "Gasto Total"
FROM Gestion.Cliente cli INNER JOIN Gestion.Cita c ON cli.ClienteID = c.ClienteID WHERE c.Estado = 'Atendida'
GROUP BY cli.Nombres, cli.Apellidos HAVING SUM(c.Total) > 50 ORDER BY "Gasto Total" DESC;
GO
```

- **Gestion.v\_ClientesRecurrentes:** Lista a los usuarios que han completado más de una visita exitosa al sistema. Ayuda a medir la tasa de retención de la marca y a separar a los clientes ocasionales de los que ya han sido fidelizados.

```
CREATE OR ALTER VIEW Gestion.v_ClientesRecurrentes AS
SELECT cli.Nombres AS "Cliente", COUNT(c.CitaID) AS "Visitas Exitosas"
FROM Gestion.Cliente cli INNER JOIN Gestion.Cita c ON cli.ClienteID = c.ClienteID WHERE c.Estado = 'Atendida'
GROUP BY cli.Nombres HAVING COUNT(c.CitaID) > 1;
GO
```

- **Gestion.v\_UltimaVisitaCliente:** Muestra el correo electrónico del cliente junto a la fecha de su última cita y los días transcurridos desde entonces. Es fundamental para



ESCUELA POLITÉCNICA NACIONAL  
FACULTAD DE INGENIERÍA DE SISTEMAS  
INGENIERÍA DE SOFTWARE

campañas de marketing dirigidas a "recuperar" clientes que no han asistido en un periodo prolongado.

```
CREATE OR ALTER VIEW Gestion.v_UltimaVisitaCliente AS
SELECT cli.Email, MAX(c.FechaHora) AS "Ultima Visita", DATEDIFF(DAY, MAX(c.FechaHora), GETDATE()) AS "Dias Ausente"
FROM Gestion.Cliente cli INNER JOIN Gestion.Cita c ON cli.ClienteID = c.ClienteID WHERE cli.Email IS NOT NULL GROUP BY
cli.Email;
GO
```

- **Gestion.v\_DistribucionDemografica:** Presenta un resumen estadístico de la base de clientes segmentado por género. Esta información es clave para ajustar el portafolio de servicios y la comunicación publicitaria según el público dominante.

```
CREATE OR ALTER VIEW Gestion.v_DistribucionDemografica AS
SELECT CASE WHEN Sexo = 'M' THEN 'Masculino' ELSE 'Femenino' END AS "Genero", COUNT(*) AS "Cantidad Clientes" FROM
Gestion.Cliente GROUP BY Sexo;
GO
```

## 12.4. Bloque 4: Análisis de Servicios

- **Gestion.v\_RankingServiciosPopulares:** Ordena el catálogo de servicios según el volumen de ventas. Permite conocer cuáles son los procedimientos favoritos de los clientes (como el corte clásico o diseño de barba) para asegurar que todas las sedes tengan insumos suficientes.

```
CREATE OR ALTER VIEW Gestion.v_RankingServiciosPopulares AS
SELECT TOP 100 PERCENT s.NombreServicio, COUNT(d.DetalleServicioID) AS "Veces Vendido"
FROM Gestion.Servicio s INNER JOIN Gestion.DetalleCitaServicio d ON s.ServicioID = d.ServicioID GROUP BY s.NombreServicio
ORDER BY "Veces Vendido" DESC;
GO
```

- **Gestion.v\_ServiciosMasRentables:** A diferencia del ranking por volumen, esta vista ordena los servicios por el dinero total generado. Ayuda a identificar servicios que, aunque no se vendan masivamente, aportan un margen de utilidad significativo a la franquicia.

```
CREATE OR ALTER VIEW Gestion.v_ServiciosMasRentables AS
SELECT TOP 100 PERCENT s.NombreServicio, SUM(s.Precio) AS "Dinero Generado"
FROM Gestion.Servicio s INNER JOIN Gestion.DetalleCitaServicio d ON s.ServicioID = d.ServicioID GROUP BY s.NombreServicio
ORDER BY "Dinero Generado" DESC;
GO
```

- **Gestion.v\_AnalisisCitasCombinadas:** Identifica las transacciones donde el cliente solicitó más de un servicio simultáneamente. Es útil para analizar el comportamiento de compra y diseñar " combos" o paquetes promocionales basados en los servicios que suelen consumirse juntos.

```
CREATE OR ALTER VIEW Gestion.v_AnalisisCitasCombinadas AS
SELECT c.CitaID, cli.Nombres AS "Cliente", COUNT(d.ServicioID) AS "Cantidad Servicios"
FROM Gestion.Cita c INNER JOIN Gestion.DetalleCitaServicio d ON c.CitaID = d.CitaID INNER JOIN Gestion.Cliente cli ON c.
ClienteID = cli.ClienteID
GROUP BY c.CitaID, cli.Nombres HAVING COUNT(d.ServicioID) > 1;
GO
```

## 12.5. Bloque 5: Auditoría y Control Temporal

- **Gestion.v\_ResumenDiarioVentas:** Proporciona un corte de caja diario, consolidando el número de citas y el monto total recaudado. Facilita la conciliación financiera y el monitoreo del flujo de efectivo día tras día.

```
CREATE OR ALTER VIEW Gestion.v_ResumenDiarioVentas AS
SELECT CAST(FechaHora AS DATE) AS "Fecha", COUNT(CitaID) AS "Citas", SUM(Total) AS "Venta Dia"
FROM Gestion.Cita WHERE Estado = 'Atendida' GROUP BY CAST(FechaHora AS DATE);
GO
```



ESCUELA POLITÉCNICA NACIONAL  
FACULTAD DE INGENIERÍA DE SISTEMAS  
INGENIERÍA DE SOFTWARE

- **Gestion.v\_PicosDeAtencion:** Analiza la afluencia de clientes distribuida por horas del día. Esta vista permite a los franquiciados optimizar los turnos del personal, reforzando la cantidad de barberos durante las horas de mayor demanda.

```
CREATE OR ALTER VIEW Gestion.v_PicosDeAtencion AS
SELECT TOP 100 PERCENT DATEPART(HOUR, FechaHora) AS "Hora del Dia", COUNT(CitaID) AS "Afluencia"
FROM Gestion.Cita GROUP BY DATEPART(HOUR, FechaHora) ORDER BY "Afluencia" DESC;
GO
```

- **Gestion.v\_CitasFuturas:** Muestra un listado de los próximos compromisos agendados que aún están en estado 'Pendiente'. Ayuda a los barberos y administradores a prepararse para la carga de trabajo de los días siguientes.

```
CREATE OR ALTER VIEW Gestion.v_CitasFuturas AS
SELECT c.FechaHora, cli.Nombres + ' ' + cli.Apellidos AS "Cliente", b.Nombres AS "Barbero Asignado"
FROM Gestion.Cita c INNER JOIN Gestion.Cliente cli ON c.ClienteID = cli.ClienteID INNER JOIN Gestion.Barbero b ON c.
BarberoID = b.BarberoID
WHERE c.FechaHora > GETDATE() AND c.Estado = 'Pendiente';
GO
```

- **Gestion.v\_PerdidasPorCancelacion:** Cuantifica el impacto económico de las citas que no llegaron a concretarse (canceladas o no asistidas). Permite visualizar cuánto dinero dejó de percibir la empresa debido al ausentismo.

```
CREATE OR ALTER VIEW Gestion.v_PerdidasPorCancelacion AS
SELECT Estado AS "Motivo Perdida", COUNT(CitaID) AS "Cantidad", ISNULL(SUM(Total), 0) AS "Dinero Perdido"
FROM Gestion.Cita WHERE Estado IN ('Cancelada', 'No Asistida') GROUP BY Estado;
GO
```

- **Gestion.v\_AuditoriaPrecios:** Compara el precio oficial del catálogo contra el precio efectivamente registrado en los detalles de la cita. Es una herramienta de control interno para detectar discrepancias o errores manuales en el cobro de servicios.

```
CREATE OR ALTER VIEW Gestion.v_AuditoriaPrecios AS
SELECT d.DetalleServicioID, s.NombreServicio, s.Precio AS [Precio Oficial], s.Precio AS [Precio Cobrado], 0.00 AS
[Diferencia]
FROM Gestion.DetalleCitaServicio d INNER JOIN Gestion.Servicio s ON d.ServicioID = s.ServicioID;
GO
```